Attorney Docket No.: 022150-000200US

PATENT APPLICATION

METHOD AND APPARATUS FOR IMAGE PROCESSING

Inventor(s): Axel K. Kloth, citizen of Germany residing at
406 Mountain Laurel Ct.
Mountain View, CA 94043

Assignee: Parimics, Inc.
13237 Paramount Dr.
Saratoga, CA 95070

Entity: Small

TOWNSEND and TOWNSEND and CREW LLP
Two Embarcadero Center, Eighth Floor
San Francisco, California 94111-3834
Tel: 650-326-2400

METHOD AND APPARATUS FOR IMAGE PROCESSING

CROSS-REFERENCES TO RELATED APPLICATIONS

[0001] The present application claims benefit of the filing date of U.S. provisional application number 60/441,076, filed on January 17, 2003, entitled "Parallel Image Computation System", the content of which is incorporated herein by reference in its entirety.

FIELD OF THE INVENTION

[0002] The present invention relates to processing of image data, and more particularly to high-speed processing of image data.

BACKGROUND OF THE INVENTION

[0003] Recent advances in image capturing techniques have brought about a need for more efficient methods for processing image data, both for real-time or near real-time applications. Conventional computational systems are becoming increasingly less effective for large scale real time image processing. Conventional computational systems are also deficient when image processing is carried out for two or more spatial dimensions.

[0004] In conventional image processing systems, images are captured and gathered on a frame-by-frame basis. Thereafter, the pixels of the stored frames are sequentially accessed by one or more CPUs from one or more DRAM subsystems. This results in systems with a limited throughput. Such conventional image processing systems are often unable to provide the required processing speed if, for example, the frame rate is high, or the required resolution is high, or the total pixel count and the color depth per pixel is high, or the number of CPU instructions per pixel is high. This processing limitation is caused, in part, by the fact that the CPUs cannot access image data fast enough from the memory (i.e., DRAM) and/or fetch code from the DRAM subsystem. Multiprocessor systems have been developed and deployed but because accesses to the DRAMs are limited due to contention and blocking, these systems are unable to process the frames at the required rates.

[0005] This problem is further compounded by the fact that the image processing requirement for a two-dimensional image with NxM pixels (typically $M = 3 \times N / 4$ or $M = 4 \times N / 5$) is a linear function of N^2. This results in either $3 \times N^2 / 4$ or $4N^2 / 5$

multiplied by the number of instructions per pixel. Accordingly, as a linear image increases in size from N_1 to N_2, the image processing time increases by a factor equal to N_1^2 / N_2^2.

[0006] Figure 1 is a simplified block of a conventional image processing system 10. Image processing system 10 is shown as including CCD 15, PCI card 20 and host mainboard 30. CCD 15 is adapted to capture the images that are digitized by Analog-to-Digital Converter (ADC) 22 of PCI card 20. The digitized image data of an entire frame is then stored in frame buffer 24, also referred to as transient image RAM 24. Thereafter, the stored frame is transferred through a bridge, such as PCI Bridge 26 into host DRAM 32 of host mainboard 30 via either host mainboard 32's CPU, or a bridge DMA controller, such as the DMA controller (not shown) disposed in PCI NorthBridge 34. The data transferred and stored in the host DRAM 32 is subsequently accessed for further processing. The architecture of image processing system 10 is relatively inexpensive and can be built with standard parts and off-the-shelf processors, however, it has a limited throughput.

[0007] The increasing complexity of manufactured products and the requirement to consistently achieve high yields when manufacturing these complex products necessitates a high quality control and product inspection standards, such as those defined by ISO 9000 standards. Such quality control systems are often required to test a product (also referred to as a unit) under static conditions as well as when the unit is in operation. This requires high frame processing rates in order to provide the capability to tag defective units so as to reduce additional testing and adjust manufacturing process parameters in real time in order to increase yield. Despite their ever-increasing CPU performance, conventional image processing systems are not fully effective at handling such high frame processing rates, partly due to the latency associated with the DRAMs.

[0008] Another technique that has been developed to reduce the cost of image processing is to integrate the image capture and gathering operations--often done via a charge coupled device (CCD)--and some computational circuitry in one integrated circuit (IC). However, this technique decreases the image quality since a significant portion of the chip area is not light sensitive and cannot capture images. These ICs (also referred to as chips) typically include one processing engine per pixel that is physically formed on the chip adjacent or in the vicinity of that pixel (i.e., co-located with the pixel). Also, because these CCD chips are formed on a DRAM-type process, the speed of the processing engine of such chips is often limited, particularly, if the processing engines were configured to be software programmable.

Moreover, the optical resolution of these chips deteriorates as the size of the co-located processing engines increases. Furthermore, these chips are only capable of capturing images in the range of the visible light, and thus fail to capture and process images generated by SONAR, RADAR, or light detection and ranging (LIDAR), and all other frequency ranges of the electromagnetic spectrum of waves.

[0009] Fig. 2 shows processing image layer 50 and 55, in accordance with conventional image processing techniques. As known to those skilled in the art, layer 50 is adapted to perform object recognition and association. Layer 55 is adapted to perform feature extraction. Because, as seen from Fig. 2, the granularity of the layers is limited, prior art image processing systems are unable to map specific image processing tasks to hardware dedicated to perform these tasks.

BRIEF SUMMARY OF THE INVENTION

[0010] In accordance with the present invention, an image processing system adapted to process image frames includes, in part, an image processing engine adapted to perform object-independent processing corresponding to a first processing layer of the image processing system, a post processing engine adapted to perform object-dependent processing corresponding to a second processing layer of the image processing system, and a processing engine adapted to perform object composition, recognition and association corresponding to a third processing layer of the image processing system. The image processing engine is further adapted to include a multitude of processors each associated with a different one of the pixels of the image frame. The post processing engine is further adapted to include an N-way symmetric multi-processing system (SMP) having disposed therein N DFT engines and N matrix multiplication engines, where N is an integer greater than 1.

[0011] In some embodiments, the multitude of processors of the image processing system corresponding to the first layer, form a massively parallel processing system that is of a systolic array type and configured as a single-instruction multiple-data system. Each of the multitude of the processors is further configured to perform object-independent processing using a unified and symmetric processing in N dimensions in space and one dimension in time.

[0012] Prior to being processed, images are captured via an image capturing device that may be a charged coupled device (CCD) and formed on a first semiconductor substrate. One

or more analog-to-digital converters may also be formed on the first semiconductor substrate to convert the captured analog image data to digital data. A realignment buffer realigns the data received from the analog-to-digital converters. The multitude of processors corresponding to the first layer are formed on a second semiconductor substrate that is separate from the first semiconductor substrate.

BRIEF DESCRIPTION OF THE DRAWINGS

[0013] Figure 1 is a simplified high-level block diagram of an image processing system, as known in the prior art.

[0014] Fig. 2 shows various layers of image processing functions, as known in the prior art.

[0015] Fig. 3 shows various layers of image processing functions, in accordance with one embodiment of the present invention.

[0016] Figure 4 is a simplified high-level block diagram of an image processing engine, in accordance with one embodiment of the present invention.

[0017] Figure 5 is a simplified high-level block diagram of each atomic processing engine disposed in the image processing engine of Fig. 4, in accordance with one embodiment of the present invention.

[0018] Figure 6 is a simplified high-level block diagram of a post processing engine (PPE), in accordance with one embodiment of the present invention.

[0019] Figure 7 is a simplified high-level block diagram of an image processing system, in accordance with one embodiment of the present invention.

DETAILED DESCRIPTION OF THE INVENTION

[0020] Fig. 3 shows various processing layers (alternatively referred to hereinbelow as layers) of an image processing system adapted to process images, in accordance with one embodiment of the present invention. The image processing system of the present invention is shown in Fig. 3 as including 5 layers, namely an object independent processing layer 105, an object dependent processing layer 110, an object composition layer 115, an object recognition layer 120, and an object association layer 125. It is understood that object composition layer 115, object recognition layer 120, and object association layer 125, in

combination, correspond to object recognition and association layer 50, as shown in Fig. 2 and hence are not described below. It is further understood that object composition layer 115, object recognition layer 120, and object association layer 125 may be combined into, for example, two layers.

[0021] In accordance with the present invention, object-independent processing layer 105, may be mapped onto (i.e., performed by) a massively parallel processor (MPP) or systolic array (shown in Fig. 5), and the object-dependent processing layer 110, may be mapped onto a symmetric multi-processor system (SMP), a uniprocessor, or an MPP. In one embodiment, the MPP may be an image processing engine (IPE), and the SMP may be a post processing engine (PPE), as described further below. Fig. 7 is a simplified high-level block diagram of an image processing system 400, in accordance with one embodiment of the present invention. As seen from Fig, 7 and described in detail below, system 400 includes, in addition to other components, an IPE 200, and a PPE 500. It is understood that object-independent processing refers to processing of pixel data that have not been associated with any objects and object-dependent processing refers to processing of data that have been associated with objects.

[0022] Fig. 4 is a simplified high-level block diagram of an IPE 200 that is adapted to perform the tasks associated with layer 105 (see Fig. 3). IPE 200 includes, in part, an array of atomic processing engines 300 (alternatively referred to hereinbelow as atomic processing elements), and a central instruction dispatch unit 210. As shown in Fig. 4, each atomic processing element (APE) is adapted to communicate with and exchange data with a neighboring APE. Each APE 300 is associated with one pixel and thus is dedicated to perform the image processing functions for that pixel. In accordance with the present invention, IPE 200 (and thus APEs 300) is formed on a semiconductor substrate that does not include any image capturing circuitry. In some embodiments, for each PPE, there may be four IPEs (i.e., IPE to PPE ratio of four) In other embodiments, this ratio may be smaller than four.

[0023] In Fig. 4, the topological distance between neighboring APEs is shown as being equal to one. In other embodiments (not shown) the topological distance between neighboring APEs may be greater than one, in which case, the neighbors may not be immediate neighbors. The neighbors, regardless of whether they are immediate neighbors or not, may have a spatial or timing relationship, or a combination thereof. In a two-dimensional

array, such as that shown in Fig 4, each APE 300 is formed so as to have eight immediate neighbors with a topological distance of one. In a three-dimensional array, each APE 300 is formed so as to have twenty-six immediate neighbors. It is understood that neighboring pixels, whether they are immediate neighbors or nor, may be assigned different weight coefficients, as required by the convolution operation.

[0024] Fig. 5 is a high-level block diagram of each APE 300. In Fig. 5 each APE 300 is shown as having has three associated registers each associated with colors red, green and blue. It is understood that in other embodiments, for frequencies falling outside the visible spectrum, such as ultraviolet or infrared, or for electromechanical phenomena, such as sound waves, APE 300 associates three different wavelengths to the respective red, green and blue registers. For example, in these embodiments, APE 300 is adapted to associate sound waves falling in the frequency range F_1 to F_2 to color red, sound waves falling in the frequency range F_2 to F_3 to color green, and sound waves falling in the frequency range F_3 to F_4 to color blue.

[0025] Referring to Fig. 5, color red is shown as having associated registers 302, 304, and 306. Color green is shown as having associated registers 308, 310, and 312. Color blue is shown as having associated registers 312, 314, and 316. Each of these registers is configured to have, e.g., 8 bits. Disposed in each APE 300 is a communication interface and vector processor 320 that is configured to communicate with other APEs. Pixel data associated with color red and corresponding to times T-2, T-1, and T are respectively stored in registers 302, 304, and 306. Pixel data associated with color green and corresponding to times T-2, T-1, and T are respectively stored in registers 308, 310, and 312. Pixel data associated with color blue and corresponding to times T-2, T-1, and T are respectively stored in registers 314, 316, and 318.

[0026] During each cycle, data is loaded into registers 306, 312, and 318 from communication interface and vector processor 320. With each cycle, new data is loaded into registers 306, 312, and 318, the data previously loaded into registers 306, 312, and 318 is shifted and stored, respectively, in registers 304, 310, and 316, and the data previously loaded into registers 304, 310, and 316 is shifted and stored, respectively, in registers 302, 308, and 314. Furthermore, during each cycle, the data in each of registers 302, 304, 306, 308, 310, 312, 314, 316, and 318 may be accessed by arithmetic logic unit and instruction sequencer (hereinafter alternatively referred to as ALU) 322 of APE 300, modified by ALU 322, and

written back to that register by ALU 322. Data shifted out of registers 302, 308, and 314 is transferred outside its respective APE, via the APE's associated communication interface and vector processor 320.

[0027] ALU 322 is further configured to transfer data to communication interface and vector processor 320. The instructions performed on the data may be transferred between ALU 322 and communication interface and vector processor 320. Communication interface and vector processor 320 receives from and delivers data to neighboring APEs via data bus 324. Communication interface and vector processor 320 as well as ALU 322 receive instructions from central instruction dispatch 210 (see Fig. 4) via instruction bus 326, and deliver status information (e.g., error messages) to central instruction dispatch 210 via status bus 328.

[0028] IPE 200 outputs data in a format that is specific to it. Data output by IPE 200 may be either an image frame that is modified, or may be a pixel representation of the image frame, or a vector representation of the objects in the image frame, or the unmodified image frame. This information can be used to generate a set of vectors describing the objects in the frame. IPE 200 is further adapted to extract features and deliver the extracted features to the PPEs for post processing.

[0029] The following is a description of one known algorithm for determining motion vectors, as performed, for example, by IPE 200 and PPE 500 of the image processing system 400 of Fig. 7. IPE 200 determines motion vectors or perimeters (also referred to hereinbelow as edge) in the form of $(a_2 - a_1, b_2 - b_1)$, where a and b refer to any scalar associated with a pixel corresponding to the perimeter of that object.

[0030] IPE 200 is also configured to determine the gradient between any two scalars of neighboring pixels. PPE 500 is configured to determine the length (norm) of vector v because it performs the operation:

$$|v| = \sqrt{(a_2 - a_1)^2 + (b_2 - b_1)^2}$$

for two-dimensional arrays and performs the operation:

$$|v| = \sqrt{(a_2 - a_1)^2 + (b_2 - b_1)^2 + (c_2 - c_1)^2}$$

for three-dimensional arrays.

[0031] As known by those skilled in the art, direction of travel and length of vector per time unit determines speed and orientation in the projected plane. The resulting motion vectors computed by IPE 200 may require to be averaged out over time by PPE 500. Therefore, multiple readings may have to be taken and averaged. Thus, linear motions are easy to detect and average out.

[0032] Object motion is determined by PPE 500 since the PPE 500 tracks the objects. As described above, pixel changes are tracked and computed by IPE 200. IPE 200 is further adapted to track pixel motion or pixel changes over time or across the space stage. Because IPE 200 unifies both time and space stage computations, IPE 200 tracks object edge motions. IPE 200 detects vectors, such as edges, and motion vectors, by sequentially extending the starting and ending point of a line. A line or a perimeter is determined by comparing the line's gradient to a predefined threshold. PPE 500 receives the perimeter data detected by IPE 200 and associates these detected edges to their respective objects.

[0033] Fig. 6 is a simplified high-level block diagram of a PPE 500, in accordance with one embodiment of the present invention, coupled to a general purpose CPU 508, and a memory 10, such as a DRAM and/or flash memory 510. PPE 500 is shown as including a load sharing demultiplexer 502, sixteen vector engines 504, and a multiplexer 506. It is understood, however, that in other embodiments, PPE 500 may include more or fewer than sixteen vector engines 504. As shown in Fig. 7, each vector engine 504 is further adapted to include a Discrete Fourier Transform (DFT) engine and a matrix multiplication engine.

[0034] Load sharing demultiplexer 502 demultiplexes data it receives from IPE 200 and distributes the received data to one of the multitude of vector engines 504. Each vector engine 504 is configured to compute linear motion, angle correction by a cosine function, compute average linear or circular motions using vector add and subtract functions, perform vector multiplication with a scalar, and perform vector rotation, i.e., a multiplication of a vector by a matrix. For motion predictions other than linear motions, the DFT engines of vector engines 504 perform discrete Fourier transform operations. In some embodiments, date related to rotating picture or linear or accelerated movement is computed and compensated for by PPE 500, and such data are valid for as long as the focus is not changed. PPE 500 of Fig. 7 is shown as including a 16-way symmetric multi-processing system (SMP), with 16 DFT engines and 16 matrix multiplication engines that convert the data related to the objects and

their motions to the format required by host CPU 426 (see Fig. 7). It is understood that in other embodiments, PPE 500 is configured to include an N-way symmetric multi-processing system (SMP), with N DFT engines and N matrix multiplication engines, where N is an integer greater than 1.

[0035] Multiplexer 506 is adapted to multiplex the data that it receives from vector engines 504 and transfers the multiplexed data to host CPU 426 (see Fig. 7). General purpose CPU 508 is used to load programming codes to vector engines 504. The DRAM of memory 508 is used to store temporary data, if necessary, from vector engines 504, and the flash memory of memory 508 is used to store programming codes used by general purpose CPU 508 and/or vector engines 504.

[0036] Since only the absolute value of the computed gradient is used, both points along the gradient--steepest incline and/or decline--are considered and used as points along a trajectory. This enables straight vectors, vectors with ragged edges along the horizontal axis, the vertical axis, and any linear combination of the two to be detected. The number of vectors delivered to the host is small compared to the number of pixels in an image since the rate of change is limited to the leading and the trailing object edge. All other pixels are static or quasi-static. Typically, an image containing $640x480$ pixels, each represented by 24 bits, is reduced to a few hundred vectors of edges or detected motions.

[0037] As described above, Fig. 7 is a simplified high-level simplified block diagram of an image processing 400, in accordance with one embodiment of the present invention. Image processing 400 is shown as including, in part, a camera unit 405, a high-speed high-performance interconnect module 410, such as a PCI-X or 3GIO card, and host mainboard 420. Interconnect module 410 is further configured to include, in part, a realignment unit and frame buffer 412, one ore more IPE 200, and one ore more PPE 500. Host mainboard 420 is further configured to include, in part, a host DRAM 416, a high performance interconnect module 424, such as a PCI-X or 3GIO NorthBridge 424, and a host CPU 426. It is understood that each camera unit 405 may include a CCD (or other image capturing devices) as well as one ore more ADCs.

[0038] Data received from camera unit 405 is transmitted to the IPE(s) 200 via realignment unit and frame buffer 412. Realignment unit and frame buffer 412 may or may not store the frame data it receives, depending on the synchronization of the multiple ADCs disposed in camera unit 405. After a frame is realigned by realignment unit and frame buffer 412 , IPE

200 starts to process the frame data. As described above, IPE 200 is adapted to detect edges, as well as differences in spatial state of the pixels and changes over time of the state of the pixels.

[0039] IPE 200 sends the detected results to PPE 500. PPE 500 is configured to compact this frame data further, if consecutive vectors are found, and to transmit this compressed frame data to the DMA controller disposed in interconnect module 424. Interconnect module 424, in turn, sends this frame data to the host DRAM 422. Host CPU 426 is configured to access the frame data stored in host DRAM 422. Therefore, in accordance with the present invention, host CPU 426 does not have to perform vector extraction, motion detection, or edge detection. Therefore, image processing system 400 has a significantly improved processing speed than conventional image processing systems. In some embodiments, image processing system 400 processes 10000 frames per second compared to most conventional image processing systems that can process 60 frames per second.

[0040] Because in accordance with the present invention, the image processing is partitioned into multiple layers, as shown in Fig. 3, an image processing system such as that shown in Fig. 7, which is partitioned to conform to these functional layers, is highly scalable and lends itself to easy integration. For example, as the feature size (e.g., transistor size) of each generation of manufactured technology reduces and scales down, the number of IPEs and PPEs that may be integrated on a single semiconductor substrate increases. Furthermore, partitioning of the image processing into multiple layers, as shown in Fig. 3, enables linear scalability of the frame size and the throughput. For example, if $4 \times N$ IPEs and N PPEs provide an inadequate frame size, then one can increase the number of IPEs to $4 \times M \times N$, and the number PPEs to $M \times N$ by just parallelizing them.

[0041] Applicant has discovered that sequential processing of image data, as performed by a conventional image processing system, results in inefficiencies and throughput limitations that the parallel image processing system of the present invention overcomes.

[0042] Attached Appendix A shows a comparison of ten cycles of how neighboring pixels are processed by a conventional image processing system vs. how similar neighboring pixels are processed by an image processing system of the present invention.

[0043] The above embodiments of the present invention are illustrative and not limitative. The invention is not limited by the type of image sensor or analog-to-digital converter. Nor is the invention limited to any specific type of process technology, e.g., CMOS, Bipolar, or

BICMOS, or otherwise that may be used to manufacture the image processing system of the present invention. The invention is not limited by the type of processor, DMA controller, memory used in the image processing system. The invention is not limited by the algorithm used to determine motion vectors, perimeters, or gradients between any two scalars. Other additions, subtractions, deletions, and modifications may be made without departing from the scope of the present disclosure as set forth in the appended claims.